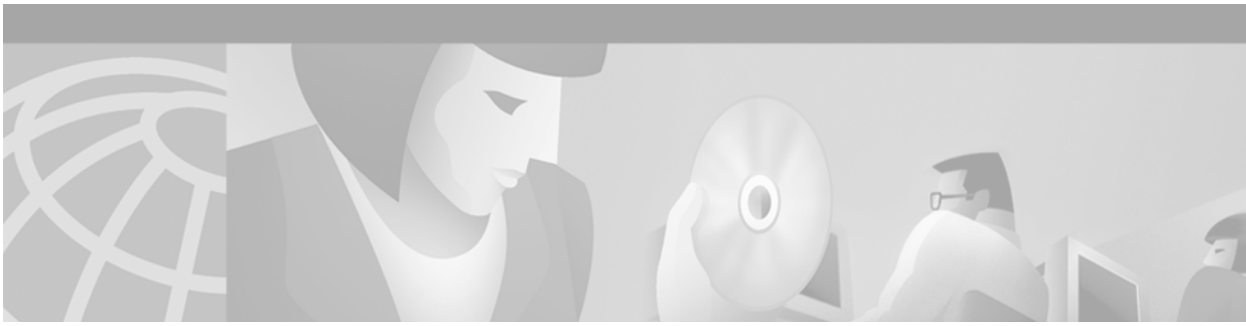


Configuring Additional File Transfer Functions

This chapter describes how to configure a router as a server, change MOP parameters, configure the router to forward extended BOOTP requests over asynchronous interfaces, and configure rcp, rsh, and FTP.

For a complete description of the file transfer function commands mentioned in this chapter, refer to the "Additional File Transfer Functions Commands" chapter in the *Configuration Fundamentals Command Reference*. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

Additional Functions Task List

To configure additional file transfer functions, perform any of the tasks in the following sections:

- Configuring a Router as a Server
- Specifying Asynchronous Interface Extended BOOTP Requests
- Configuring a Router to Use rsh and rcp
- Configuring a Router to Use FTP Connections

Configuring a Router as a Server

It is too costly and inefficient to have a machine which only acts as server on every network segment. However, when you do not have a server on every segment, your network operations can incur enormous time delays across network segments. You can configure a router to serve as a RARP or TFTP server to reduce costs and time delays in your network while allowing you to use your router for its regular functions.

Typically, a router that is configured as a server provides other routers with operating system images from its Flash memory. You can also configure the router to respond to other types of service requests, such as Reverse Address Resolution Protocol (RARP) requests.

To configure the router as a server, perform any of the tasks in the following sections. The tasks are not mutually exclusive.

- Configuring a Router as a TFTP Server
- Configuring a Router as a RARP Server

In addition, you can configure the Cisco IOS software to forward extended BOOTP requests over asynchronous interfaces. Refer to the "Configuring SLIP and PPP" chapter of the *Dial Solutions Configuration Guide* for more information.

Configuring a Router as a TFTP Server

As a TFTP server host, the router responds to TFTP Read Request messages by sending a copy of the system image contained in ROM or one of the system images contained in Flash memory to the requesting host. The TFTP Read Request message must use one of the filenames that are specified in the configuration.

Note For the Cisco 7000 family, the filename used must represent a software image that is present in Flash memory. If no image resides in Flash memory, the client router will boot the server's ROM image as a default.

Flash memory can be used as a TFTP file server for other routers on the network. This feature allows you to boot a remote router with an image that resides in the Flash server memory.

With Cisco IOS Release 11.0, Cisco 7000 family allow you to specify one of the different Flash memory devices (**bootflash:**, **slot0:**, **slot1:**, **slavebootflash:**, **slaveslot0:**, or **slaveslot1:**) as the TFTP server.

In the description that follows, one Cisco 7000 router is referred to as the *Flash server*, and all other routers are referred to as *client routers*. Example configurations for the Flash server and client routers include commands as necessary.

To configure a router as a TFTP server, perform the tasks in the following sections:

- Performing Prerequisite Tasks
- Configuring the Server
- Configuring the Client Router

Performing Prerequisite Tasks

The server and client router must be able to reach each other before the TFTP function can be implemented. Verify this connection by pinging between the server and client router (in either direction) with the **ping** command.

An example use of the **ping** command is as follows:

```
Router# ping 172.16.101.101
```

In this example, the Internet Protocol (IP) address of 172.16.101.101 belongs to the client router. Connectivity is indicated by a series of exclamation points (!), while a series of periods (.) plus *[timed out]* or *[failed]* indicates no connection. If the connection fails, reconfigure the interface, check the physical connection between the Flash server and client router, and ping again.

After you verify the connection, ensure that a TFTP-bootable image is present on the server. This is the system software image the client router will boot. Note the name of this software image so you can verify it after the first client boot.

Caution For full functionality, the software image sent to the client must be the same type as the ROM software installed on the client router. For example, if the server has X.25 software, and the client does not have X.25 software in ROM, the client will not have X.25 capabilities after booting from the server's image in Flash memory.

Configuring the Server

To specify TFTP server operation, use the following commands in configuration mode:

	Command	Purpose
Step 1	**configure terminal**	Enters global configuration mode from the terminal.
Step 2	**tftp-server flash** [*partition-number*:]*filename1* [**alias** *filename2*] [*access-list-number*] or **tftp-server flash** *device*:*filename* (Cisco 7000 family only) or **tftp-server flash** [*device*:][*partition-number*:]*filename* (Cisco 1600 series and Cisco 3600 series only) or **tftp-server rom alias** *filename1* [*access-list-number*]	Specifies the system image to send in response to Read Requests. You can enter multiple lines to specify multiple images.
Step 3	**end**	Exits configuration mode.
Step 4	**copy running-config startup-config**	Saves the configuration file to your startup configuration.

The TFTP session can sometimes fail. TFTP generates the following special characters to help you determine why a TFTP session fails:

- An "E" character indicates that the TFTP server received an erroneous packet.
- An "O" character indicates that the TFTP server received an out-of-sequence packet.
- A period (.) indicates a timeout.

For diagnosing any undue delay in the transfer, the output is useful. For troubleshooting procedures, refer to the *Internetwork Troubleshooting Guide* publication.

In the following example, the system can use TFTP to send copies of the Flash memory file *version-10.3* in response to a TFTP Read Request for that file. The requesting host is checked against access list 22.

```
tftp-server flash version-10.3 22
```

In the following example, the system can use TFTP to send a copy of the ROM image *gs3-k.101* in response to a TFTP Read Request for the *gs3-k.101* file:

```
tftp-server rom alias gs3-k.101
```

The following example a router to send a copy of the file *gs7-k.9.17* in Flash memory in response to a TFTP Read Request. The client router must reside on a network specified by access list 1. Thus, in the example, the any clients on network 172.16.101.0 are permitted access to the file.

```
Server# configure terminal
Enter configuration commands, one per line. End with CTRL/Z
Server(config)# tftp-server flash gs7-k.9.17 1
Server(config)# access-list 1 permit 172.16.101.0 0.0.0.255
Server(config)# end
Server# copy running-config startup-config
[ok]
Server#
```

Configuring the Client Router

Configure the client router to first load a system image from the server. As a backup, configure the client router to then load its own ROM image if the load from a server fails. To configure the client router, use the following commands beginning in privileged EXEC mode:

	Command	Purpose
Step 1	**configure terminal**	Enters global configuration mode from the terminal.
Step 2	**no boot system**	Removes all previous **boot system** statements from the configuration file.
Step 3	**boot system** [**tftp**] *filename* [*ip-address*]	Specifies that the client router load a system image from the server.
Step 4	**boot system rom**	Specifies that the client router loads its own ROM image if the load from a server fails.
Step 5	**config-register** *value*	Sets the configuration register to enable the client router to load a system image from a network server.
Step 6	**end**	Exits global configuration mode.
Step 7	**copy running-config startup-config**	Saves the configuration file to your startup configuration.
Step 8	**reload**	Reloads the router to make your changes take effect.
Step 9		(Router reboots.)
Step 10	**show version**	Verifies that the client router booted the correct image from the TFTP server.

Caution Using the **no boot system** command, as in the following example, will invalidate *all* other boot system commands currently in the client router system configuration. Before proceeding, determine whether the system configuration stored in the client router should first be saved (uploaded) to a TFTP file server so you have a backup copy.

The following example shows how to configure a router to use a TFTP server:

```
Client# configure terminal
Enter configuration commands, one per line. End with CTRL/Z
Client(config)# no boot system
Client(config)# boot system gs7-k.9.17 172.31.111.111
Client(config)# boot system rom
Client(config)# config-register 0x010F
Client(config)# end
Client# copy running-config startup-config
[ok]
Client# reload
```

In this example, the **no boot system** command invalidates all other **boot system** commands currently in the configuration memory, and any **boot system** commands entered after this command will be executed first. The second command, **boot system** *filename address*, tells the client router to look for the file *gs7-k.9.17* on the TFTP server with an IP address of 172.31.111.111. Failing this, the client router will boot from its system ROM in response to the **boot system rom** command, which is included as a backup in case of a network problem. The **copy running-config startup-config** command copies the configuration to the startup configuration, and the **reload** command boots the system.

Caution The system software (*gs7-k.9.17* in the example) to be booted from the server (172.31.111.111 in the example) must reside in Flash memory on the server. If it is not in Flash memory, the client router will boot the server's system ROM.

The following example shows sample output of the **show version** command after the router has rebooted:

```
Client> show version
GS Software (GS7), Version 9.1.17
Copyright (c) 1986-1992 by cisco Systems, Inc.
Compiled Wed 21-Oct-92 22:49

System Bootstrap, Version 4.6(0.15)

Current date and time is Thu 10-22-1992 13:15:03
Boot date and time is Thu 10-22-1992 13:06:55
env-chassis uptime is 9 minutes
System restarted by power-on
System image file is "gs7-k.9.17", booted via tftp from 172.31.111.111

RP1 (68040) processor with 16384K bytes of memory.
X.25 software.
Bridging software.
1 Switch Processor.
1 EIP controller (6 Ethernet).
6 Ethernet/IEEE 802.3 interface.
128K bytes of non-volatile configuration memory.
4096K bytes of flash memory on embedded flash (in RP1).
Configuration register is 0x010F
```

The important information in this example is contained in the first line "GS Software..." and in the line that begins "System image file...." The "GS Software..." line shows the version of the operating system in the client router's RAM. The "System image file...." line show the filename of the system image loaded from the TFTP server.

Configuring a Router as a RARP Server

Reverse Address Resolution Protocol (RARP) is a protocol in the TCP/IP stack that provides a method for finding IP addresses based on MAC (physical) addresses. This functionality is the reverse of broadcasting Address Resolution Protocols (ARPs), through which a host can dynamically discover the MAC-layer address corresponding to a particular IP network-layer address. RARP makes diskless booting of various systems possible (for example, diskless workstations that do not know their IP addresses when they boot, such as Sun workstations or PCs on networks where the client and server are on separate subnets). RARP relies on the presence of a RARP server with cached table entries of MAC-layer-to-IP address mappings.

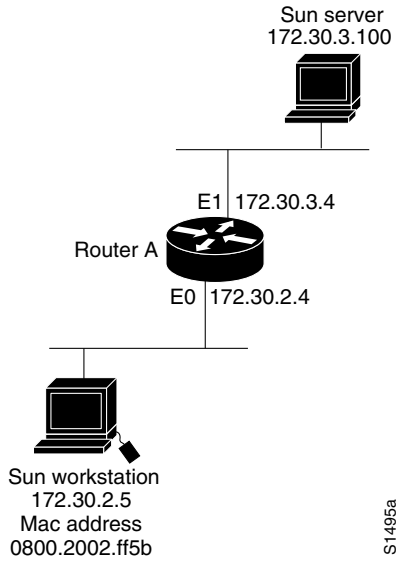
You can configure a Cisco router as a RARP server. This feature enables the Cisco IOS software to answer RARP requests.

To configure the router as a RARP server, use the following command in interface configuration mode:

Command	Purpose
ip rarp-server *ip-address*	Configures the router as a RARP server.

Figure 16 illustrates a network configuration in which a router is configured to act as a RARP server.

Figure 16 Configuring a Router as a RARP Server



Router A has the following configuration:

```
! Allow the router to forward broadcast portmapper requests
ip forward-protocol udp 111
! Provide the router with the IP address of the diskless sun
arp 172.30.2.5 0800.2002.ff5b arpa
interface ethernet 0
! Configure the router to act as a RARP server, using the Sun Server's IP
! address in the RARP response packet.
ip rarp-server 172.30.3.100
! Portmapper broadcasts from this interface are sent to the Sun Server.
ip helper-address 172.30.3.100
```

The Sun client and server's IP addresses must use the same major network number because of a limitation with the current SunOS *rpc.bootparamd* daemon.

In the following example, an access server is configured to act as a RARP server.

```
! Allow the access server to forward broadcast portmapper requests
 ip forward-protocol udp 111
! Provide the access server with the IP address of the diskless sun
 arp 172.30.2.5 0800.2002.ff5b arpa
 interface ethernet 0
! Configure the access server to act as a RARP server, using the Sun Server's
! IP address in the RARP response packet.
 ip rarp-server 172.30.3.100
! Portmapper broadcasts from this interface are sent to the Sun Server.
 ip helper-address 172.30.3.100
```

Specifying Asynchronous Interface Extended BOOTP Requests

The Boot Protocol (BOOTP) server for asynchronous interfaces supports the extended BOOTP requests specified in RFC 1084. The following command is useful in conjunction with using the auxiliary port as an asynchronous interface.

To configure extended BOOTP requests for asynchronous interfaces, use the following command in global configuration mode:

Command	Purpose
async-bootp *tag* [:*hostname*] *data*	Configures extended BOOTP requests for asynchronous interfaces.

You can display the extended BOOTP requests by using the following command in EXEC mode:

Command	Purpose
show async-bootp	Shows parameters for BOOTP requests.

Configuring a Router to Use rsh and rcp

Remote shell (rsh) gives users the ability to execute commands remotely. Remote copy (rcp) allows users to copy files to and from a file system residing on a remote host or server on the network. Cisco's implementation of rsh and rcp interoperates with the industry standard implementations.

Note Cisco uses the abbreviation RCMD (Remote Command) to indicate both rsh and rcp

The following tasks are covered in this section:

- Specifying the Source Interface for Outgoing RCMD Communications
- Disabling DNS Reverse Lookup for RCMD
- Configuring a Router to Use rsh
- Configuring a Router to Use rcp

Specifying the Source Interface for Outgoing RCMD Communications

You can specify the source interface for RCMD (rsh and rcp) communications. For example, the router can be configured so that RCMD connections use the loopback interface as the source address of all packets leaving the router. To specify the interface associated with RCMP communications, use the following command in global configuration mode:

Command	Purpose
ip rcmd source-interface *interface-id*	Specifies the interface address that will be used to label all outgoing rsh and rcp traffic.

Specifying the source-interface is most commonly used to specify a loopback interface. This allows you to associate a permanent IP address with RCMD communications.

Having a permanent IP address is useful for session identification (remote device can consistently idendify the origin of packets for the session). A "well-known" IP address can also be used for security purposes, as you can then create access lists on remote devices which include the address.

Disabling DNS Reverse Lookup for RCMD

As a basic security check, the Cisco IOS software does a reverse lookup of the client IP address using DNS. This check is performed using a host authentication process.

When enabled, the system records the address of the requesting client. That address is mapped to a host name using DNS. Then a DNS request is made for the IP address for that host name. The IP address received is then checked against the original requesting address. If the address does not match with any of the addresses received from DNS, the RCMD request will not be serviced.

This reverse lookup is intended to help protect against "spoofing." However, please note that the process only confirms that the IP address is a valid routable address; it is still possible for a hacker to spoof the valid IP address of a known host.

This feature is enabled by default. You can disable the DNS check for RCMD (rsh and rcp) access using the the following command in global configuration mode:

Command	Purpose
no ip rcmd domain-lookup	Disables Domain Name Service (DNS) lookup for rsh and rcp communications.

Configuring a Router to Use rsh

You can use rsh to execute commands on remote systems to which you have access. When you issue the rsh command, a shell is started on the remote system. The shell allows you to execute commands on the remote system without having to log in to the target host.

You do not need to connect to the system, router, or access server and then disconnect after you execute a command if you use rsh. For example, you can use rsh to remotely look at the status of other devices *without* connecting to the target device, executing the command, and then disconnecting. This capability is useful for looking at statistics on many different routers.

Maintaining rsh Security

To gain access to a remote system running rsh, such as a UNIX host, an entry must exist in the system's *.rhosts* file or its equivalent identifying you as a user who is authorized to execute commands remotely on the system. On UNIX systems, the *.rhosts* file identifies users who can remotely execute commands on the system.

You can enable rsh support on a router to allow users on remote systems to execute commands. However, our implementation of rsh does not support an *.rhosts* file. Instead, you must configure a local authentication database to control access to the router by users attempting to execute commands remotely using rsh. A local authentication database is similar to a UNIX *.rhosts* file. Each entry that you configure in the authentication database identifies the local user, the remote host, and the remote user.

Configuring the Router to Allow Remote Users to Execute Commands Using rsh

To configure the router as an rsh server, use the following commands in global configuration mode:

	Command	Purpose
Step 1	**ip rcmd remote-host** *local-username* {*ip-address* | *host*} *remote-username* [**enable** [*level*]]	Creates an entry in the local authentication database for each remote user who is allowed to execute rsh commands.
Step 2	**ip rcmd rsh-enable**	Enables the software to support incoming rsh commands.

To disable the software from supporting incoming rsh commands, use the **no ip rcmd rsh-enable** command.

Note When support of incoming rsh commands is disabled, you can still issue an rsh command to be executed on other routers that support the remote shell protocol and on UNIX hosts on the network.

The following example shows how to add two entries for remote users to the authentication database, and enable a router to support rsh commands from remote users:

```
ip rcmd remote-host Router1 172.16.101.101 rmtnetad1
ip rcmd remote-host Router1 172.16.101.101 netadmin4 enable
ip rcmd rsh-enable
```

The users, named *rmtnetad1* and *netadmin4*, are both on the remote host at IP address 172.16.101.101. Although both users are on the same remote host, you must include a unique entry for each user. Both users are allowed to connect to the router and remotely execute rsh commands on it after the router is enabled for rsh. The user named *netadmin4* is allowed to execute privileged EXEC mode commands on the router. Both authentication database entries give the router's host name *Router1* as the local username. The last command enables the router for to support rsh commands issued by remote users.

Executing Commands Remotely Using rsh

You can use rsh to execute commands remotely on network servers that support the remote shell protocol. To use this command, the *.rhosts* files (or equivalent files) on the network server must include an entry that permits you to remotely execute commands on that host.

If the remote server has a directory structure, as do UNIX systems, the rsh command that you issue is remotely executed from the directory of the account for the remote user that you specify through the **/user** *username* keyword and argument pair.

If you do not specify the **/user** keyword and argument, the Cisco IOS software sends a default remote username. As the default value of the remote username, the software sends the remote username associated with the current TTY process, if that name is valid. If the TTY remote username is invalid, the software uses the router host name as the both the remote and local usernames.

To execute a command remotely on a network server using rsh, use the following commands in user EXEC mode:

	Command	Purpose
Step 1	**enable** [*password*]	Enters privileged EXEC mode.
Step 2	**rsh** {*ip-address* | *host*} [**/user** *username*] *remote-command*	Executes a command remotely using rsh.

The following example executes the "ls -a" command in the home directory of the user sharon on mysys.cisco.com using rsh:

```
Router# enable
Router# rsh mysys.cisco.com /user sharon ls -a
.
..
.alias
.cshrc
.emacs
.exrc
.history
.login
.mailrc
.newsrc
.oldnewsrc
.rhosts
.twmrc
.xsession
jazz
Router#
```

Configuring a Router to Use rcp

The rcp copy commands rely on the rsh server (or daemon) on the remote system. To copy files using rcp, you do not need to create a server for file distribution, as you do with TFTP. You need only to have access to a server that supports the remote shell (rsh). (Most UNIX systems support rsh.) Because you are copying a file from one place to another, you must have read permission on the source file and write permission on the destination file. If the destination file does not exist, rcp creates it for you.

Although our rcp implementation emulates the functions of the UNIX rcp implementation—copying files among systems on the network—our command syntax differs from the UNIX rcp command syntax. Our rcp support offers a set of copy commands that use rcp as the transport mechanism. These rcp copy commands are similar in style to our TFTP copy commands, but they offer an alternative that provides faster performance and reliable delivery of data. These improvements are possible because the rcp transport mechanism is built on and uses the Transmission Control Protocol/Internet Protocol (TCP/IP) stack, which is connection-oriented. You can use rcp commands to copy system images and configuration files from the router to a network server and vice versa.

You can also enable rcp support to allow users on remote systems to copy files to and from the router.

Configuring the Router to Accept rcp Requests from Remote Users

To configure the Cisco IOS software to support incoming rcp requests, use the following commands in global configuration mode:

	Command	Purpose
Step 1	**ip rcmd remote-host** *local-username* {*ip-address* | *host*} *remote-username* [**enable** [*level*]]	Create an entry in the local authentication database for each remote user who is allowed to execute rcp commands.
Step 2	**ip rcmd rcp-enable**	Enable the software to support incoming rcp requests.

To disable the software from supporting incoming rcp requests, use the **no ip rcmd rcp-enable** command.

Note When support for incoming rcp requests is disabled, you can still use the rcp commands to copy images from remote servers. The support for incoming rcp requests is distinct from its ability to handle outgoing rcp requests.

The following example shows how to add two entries for remote users to the authentication database and then enable the software to support remote copy requests from remote users. The users, named *netadmin1* on the remote host at IP address 172.16.15.55 and *netadmin3* on the remote host at IP address 172.16.101.101, are both allowed to connect to the router and remotely execute rcp commands on it after the router is enabled to support rcp. Both authentication database entries give the host name *Router1* as the local username. The last command enables the router to support for rcp requests from remote users.

```
ip rcmd remote-host Router1 172.16.15.55 netadmin1
ip rcmd remote-host Router1 172.16.101.101 netadmin3
ip rcmd rcp-enable
```

Configuring the Remote to Send rcp Requests

The rcp protocol requires a client to send a remote username on each rcp request to a server. When you copy a configuration file from a server to the router using rcp, the Cisco IOS software sends the first valid username in the following list:

1. The username set by the **ip rcmd remote-username** command, if the command is configured.
2. The remote username associated with the current TTY (terminal) process. For example, if the user is connected to the router through Telnet and was authenticated through the **username** command, the router software sends the Telnet username as the remote username.

Note For Cisco, TTYs are commonly used in access servers. The concept of TTY originated with UNIX. For UNIX systems, each physical device is represented in the file system. Terminals are called *TTY devices*, which stands for *teletype*, the original UNIX terminal.

3. The router host name.

For **boot** commands using rcp, the software sends the router host name; you cannot explicitly configure the remote username.

For the rcp copy request to execute successfully, an account must be defined on the network server for the remote username.

If you are writing to the server, the rcp server must be properly configured to accept the rcp write request from the user on the router. For UNIX systems, you must add an entry to the *.rhosts* file for the remote user on the rcp server. For example, if the router contains the following configuration lines.

```
hostname Rtr1
ip rcmd remote-username User0
```

and the router's IP address translates to Router1.company.com, then the *.rhosts* file for User0 on the rcp server should contain the following line:

```
Router1.company.com Rtr1
```

Refer to the documentation for your rcp server for more details.

If the server has a directory structure, the configuration file or image is written or copied relative to the directory associated with the remote username on the server. Use the **ip rcmd remote-username** command to specify which directory on the server to use. For example, if the system image resides in the home directory of a user on the server, you can specify that user's name as the remote username.

If you copy the configuration file to a personal computer used as a file server, the computer must support rsh.

To override the default remote username sent on rcp requests, use the following commands starting in privileged EXEC mode:

	Command	Purpose
Step 1	**configure terminal**	Enters global configuration mode from the terminal
Step 2	**ip rcmd remote-username** *username*	Specifies the remote username.

To remove the remote username and return to the default value, use the **no ip rcmd remote-username** command.

Configuring a Router to Use FTP Connections

You configure a router to transfer files between systems on the network using the Internet File Transfer Protocol (FTP). With the Cisco IOS implementation of FTP, you can set the following features:

- Passive-mode FTP
- User name
- Password
- IP address

FTP Configuration Task List

Use the instructions in the section "Configuring FTP Connections" to configure FTP on a router.

Configuring FTP Connections

To configure FTP connections on a router, use the following commands in global configuration mode:

Command	Purpose
ip ftp username *string*	Specifies the user name to be used for the FTP connection.
ip ftp password [*type*] *password*	Specifies the password to be used for the FTP connection.
ip ftp passive or **no ip ftp passive**	Configures the router to only use passive-mode FTP connections. Allows all types of FTP connections (default).
ip ftp source-interface *interface*	Specifies the source IP address for FTP connections.

The following example demonstrates how to capture a core dump using the Cisco IOS FTP feature. The router accesses a server at IP address 192.168.10.3 with login name zorro and password sword. The default passive-mode FTP is used, and the server is accessed using Token Ring interface to1 on the router where the core dump will occur:

```
ip ftp username zorro
ip ftp password sword
ip ftp passive
ip ftp source-interface to1
exception protocol ftp
! This command allows the core-dump code to use FTP rather than TFTP or RCP
exception dump 192.168.10.3
! This command creates the core dump in the event the system at IP
  address 192.168.10.3 crashes
```